Coding Bootcamp

Weekly Project 1 - Individual

Definition

You are required to build a console application which accepts two arguments, the first defining a path to a text file (.txt) while the second should be one of the following commands:

- `wc`: when the user passes the particular string as the second argument, the application will count and display the total number of words found in the text file.
- `find`: when the user passes the particular string as the second argument, the application will ask the user to enter a word to search for in the text file. The application should keep count of all occurrences and when the whole text has been parsed, it should display the result to the user.

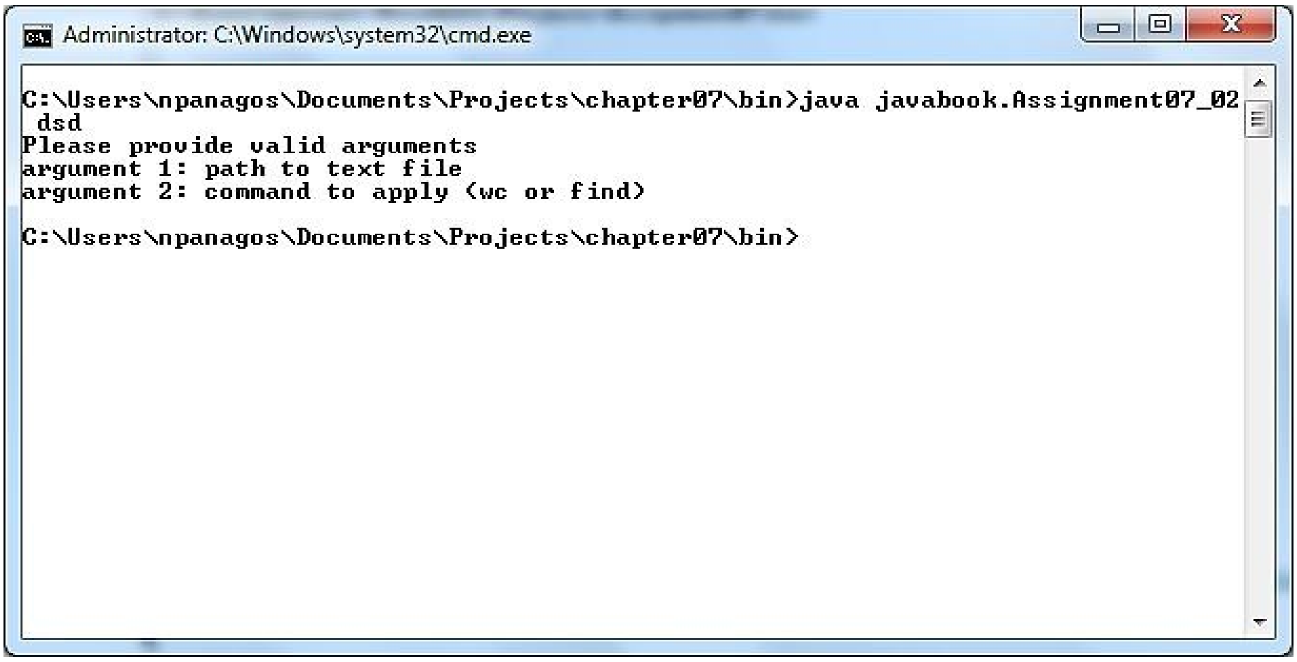
After successful completion of any of the aforementioned commands, the result should be appended to a text file named "*wordcount.txt*", in the following format:

```
Date          Time        Command     file          Result
for example:
27/06/2018    20:13:08    wc          clouds.txt    2137
27/06/2018    20:20:16    find        clouds.txt    cloud:39
```

The file should be created during the first execution of the program, at the current directory (the directory from where the application is run).

In order to check the correctness of your application, use as input the text file "*clouds.txt*", which you can find in the folder "*test_file"* on CB5 Afternoon Teams. The exact same file was used in the following application executions.

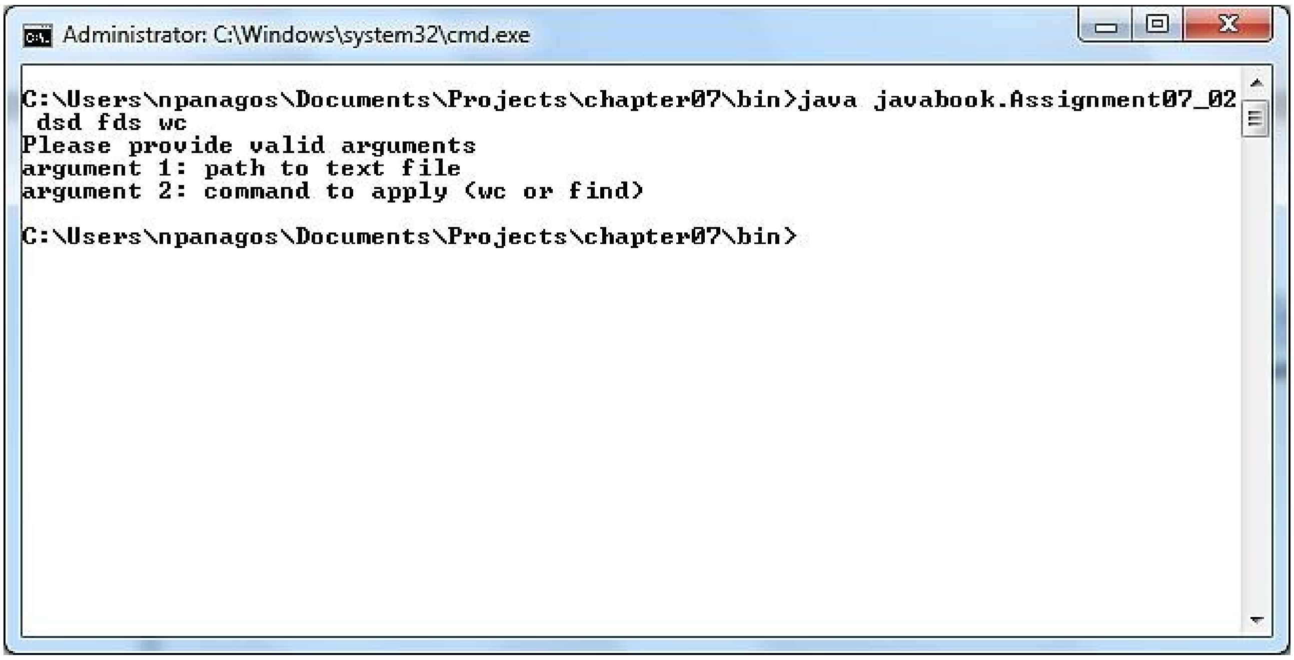
1st execution: The user runs the program passing less than 2 arguments.



```
C:\Users\npanagos\Documents\Projects\chapter07\bin>java javabook.Assignment07_02
 dsd
Please provide valid arguments
argument 1: path to text file
argument 2: command to apply (wc or find)

C:\Users\npanagos\Documents\Projects\chapter07\bin>
```

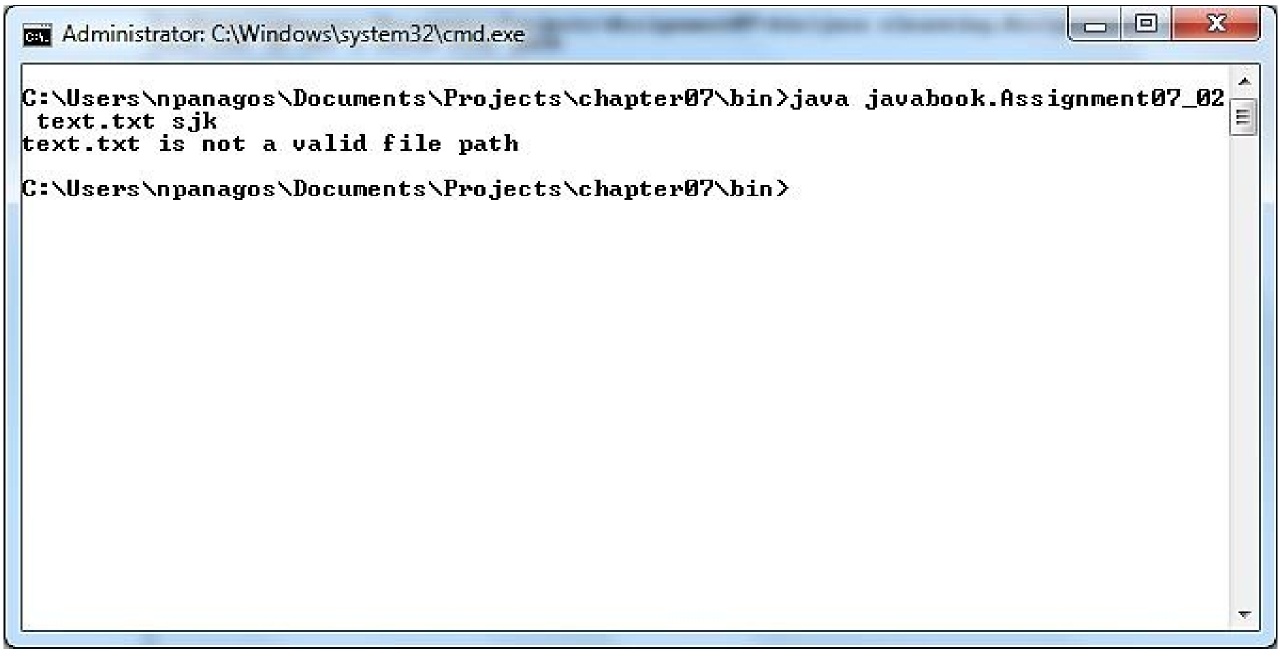
2nd execution: The user runs the program passing more than 2 arguments.



```
C:\Users\npanagos\Documents\Projects\chapter07\bin>java javabook.Assignment07_02
 dsd fds wc
Please provide valid arguments
argument 1: path to text file
argument 2: command to apply (wc or find)

C:\Users\npanagos\Documents\Projects\chapter07\bin>
```

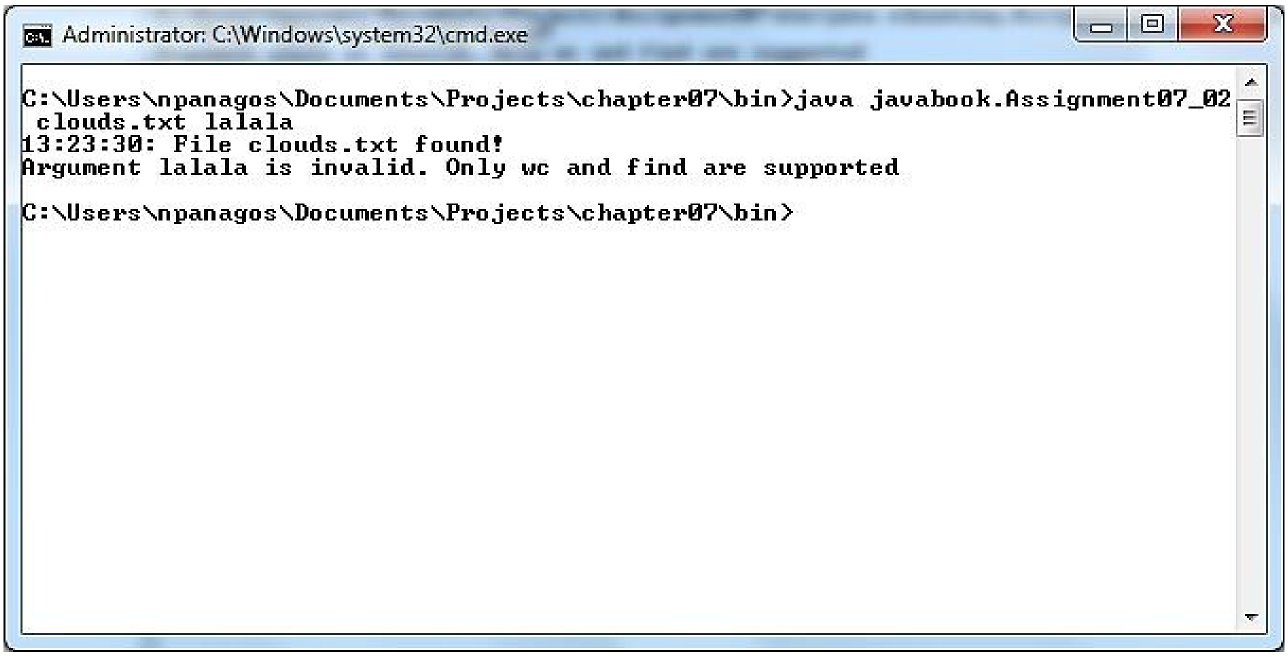
3rd execution: The user runs the program passing 2 arguments, but the first one does not correspond to a valid file path



```
C:\Users\npanagos\Documents\Projects\chapter07\bin>java javabook.Assignment07_02
 text.txt sjk
text.txt is not a valid file path

C:\Users\npanagos\Documents\Projects\chapter07\bin>
```

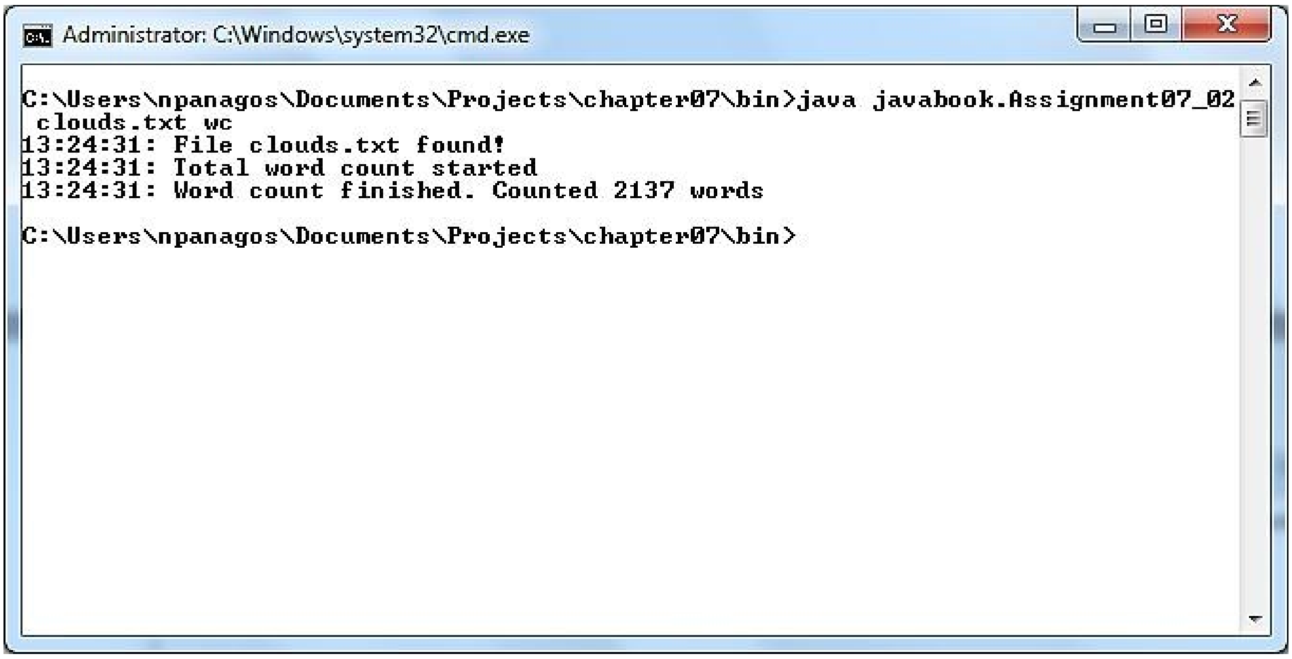
4th execution: The user runs the program passing 2 arguments, the first corresponding to a valid text file path while the second one being an invalid command (that is not `wc` or `find`).



```
C:\Users\npanagos\Documents\Projects\chapter07\bin>java javabook.Assignment07_02
 clouds.txt lalala
13:23:30: File clouds.txt found!
Argument lalala is invalid. Only wc and find are supported

C:\Users\npanagos\Documents\Projects\chapter07\bin>
```

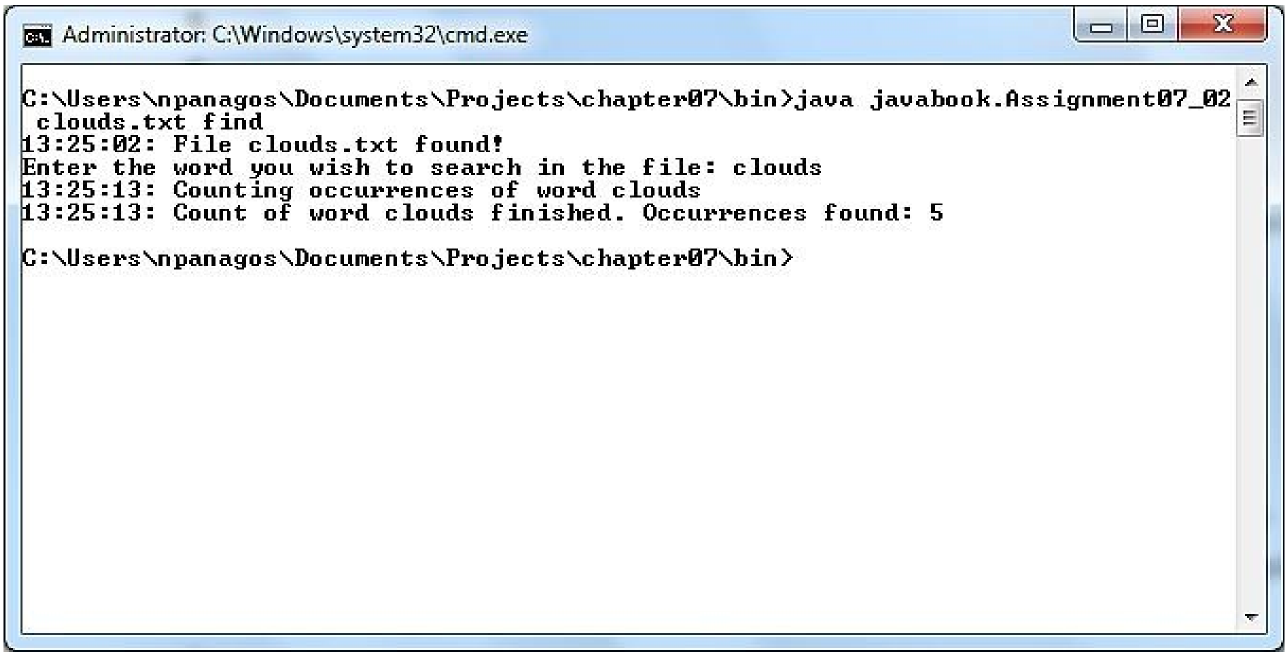
5th execution: The user runs the program passing 2 arguments, one translating to a valid text file path and the second being the `wc` command.



```
C:\Users\npanagos\Documents\Projects\chapter07\bin>java javabook.Assignment07_02
 clouds.txt wc
13:24:31: File clouds.txt found!
13:24:31: Total word count started
13:24:31: Word count finished. Counted 2137 words

C:\Users\npanagos\Documents\Projects\chapter07\bin>
```

6th execution: The user runs the program passing 2 arguments, one translating to a valid text file path and the second being the `find` command.



```
C:\Users\npanagos\Documents\Projects\chapter07\bin>java javabook.Assignment07_02
 clouds.txt find
13:25:02: File clouds.txt found!
Enter the word you wish to search in the file: clouds
13:25:13: Counting occurrences of word clouds
13:25:13: Count of word clouds finished. Occurrences found: 5

C:\Users\npanagos\Documents\Projects\chapter07\bin>
```

Guidelines

- There are many ways to solve this particular problem and you are free to choose the one you like the most so long as you are getting the same output as the one in the screenshots. To implement `find` you may have to use `Pattern` and `Matcher` classes in combination with a regular expression.
- The easiest way to read an input file is with the help of a `BufferedReader` and the `readline()` method in a loop. Each line of text can be then broken down with the help of a `Scanner` object.
- The `find` command has a similar operation to the one of the popular browsers, that is it matches all occurrences, even if the character sequence we are looking for is part of a bigger word. For example, if the user chose to count the occurrences of the word 'on' in the following text, a correct implementation should count all character sequences highlighted in yellow color.

  Introduction

  Everyone has an opinion on what is cloud computing. It can be the ability to rent a

  That makes your job easier as you are not required to write a complex regular expression.